**FP-Aufgaben 08**

Aufgabe 1

Welche Fehler finden Sie in dem folgenden Programm?

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

* 1. Pattern für die Funktion **plus** ist nicht vollständig
* 2. Man versucht eine Liste von Zahl per **(==)** mit einer leeren Liste zu vergleichen, Zahl muss ableiten von **Eq** also **"deriving (Eq)"**, muss bei der Definition von **Zahl** stehen.
* 3. Fehler ist, dass versucht wird den Typen **Zahl** mit einer Variablen **x** per **(:)** zu einer List zu Konkatenieren
* 4. Fehler ist, dass die Funktion **(+)** nicht für Datentypen **Zahl** definiert ist. Was wohl benutzt werden soll ist die plus Funktion.

Aufgabe 2

a) Definieren Sie einen Typ für Bäume.

data Tree v

= EmptyTree

| LeafTree v

| BranchTree v (Tree v)

b) Definieren Sie einen Typ für Binärbäume.

data BinaryTree v

= Empty

| Branch v (BinaryTree v) (BinaryTree v)

deriving (Show)

c) Definieren Sie eine Funktion, die kontrolliert, ob eine eingegebene Zahl in dem Baum (bzw. Binärbaum) existiert.

isInTree :: Eq v => v -> BinaryTree v -> Bool

isInTree value Empty = False

isInTree value (Branch val leftTree rightTree) =

( value == val

|| isInTree value leftTree

|| isInTree value rightTree

)

d) Definieren Sie eine Funktion, die einen Binärbaum in einer Liste konvertiert.

treeToList :: BinaryTree v -> [v]

treeToList Empty = []

treeToList (Branch val leftTree rightTree) =

(treeToList leftTree) ++ [val] ++ (treeToList rightTree)

e) Definieren Sie eine Funktion, die eine sortierte Liste in einem Binärbaum konvertiert.

listToTree :: [v] -> BinaryTree v

listToTree [] = Empty

listToTree xs =

Branch

(xs !! half)

(listToTree (take half xs))

(listToTree (drop (half + 1) xs))

where

len = length xs

half = len `div` 2